Networks Security & Distrubuted Systems (NSDS'2015)
25 - 26 November , Baghdad IRAQ

Proposed Method to Prevent SQL Injection Attack

Makera M Aziz
Business Management Department
Ishik University
Erbil, Iraq
Makera.aziz@Ishik.edu.iq

Dena Rafaa Ahmed
Computer Science Department
Bayan University
Erbil, Iraq
dina.rafaa1986@gmail.com

***Abstract*— the internet and its websites have huge using these days. These webs may have sensitive and secret information like military information, financial information and other important information that transfer through the networks. Only some people have the authorization to see and access this information. So information has to transfer in secret environment. SQL injection represents one of the most important things that thread these webs. In which unauthorized people can access to the data and information. This paper introduces a method that can be used to prevent SQL injection by converting the user input to static string, use this string as user input and compared with the database attributes that need to compare with, during the runtime. The goal behind converting the input to a string is to make user input as a single unit (one token) that cannot use as a SQL query statement. The system will call the database attribute in such away in which user cannot access to the sql statement to do the injection. And the sql query will be empty from any input tools that can use by user to injects the SQL.**

***Index Terms*—SQL Injection, Network security, database security**

I. INTRODUCTION

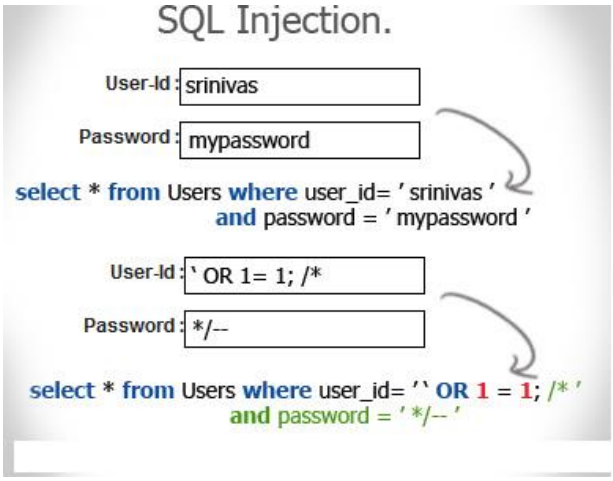
Today internet is considered as a large source of information. Some of this information is sensitive information and use in financial fraud, online banking, online shopping and cyber terrorism. This information it has to be transferred in secret environment. Application and internet web are vulnerable to attack from the hacker. SQL injection represents one of the most important threads for these applications. Injection attacks are one of the top 10 threads for application security. The user how injects the scale of the web can get the complete access to the web database. In SQL injection the web accepts the user inputs and uses them to reform the SQL query at the run time. During attack, user inputs SQL query segment as an input which can use for the different database request. The attacker also can use the SQL injection to attack the network. [1]. The hackers can use this type of attack to attack the host of the web. [2]. The SQL injection can occur in a SQL statement that does not modify data like SELECT and also with the statement that use to modify data like INSERT, UPDATE and DELETE statement. [3].

Many Web applications do not confirm the information on the types, allowing an attacker to input malicious codes that are implemented on a back-end database. Then, the attacker can apply automated tools to collect information from tables and columns in the database. Furthermore, the attacker can compromise the database by injecting more malicious codes, potentially making a database server to download other programs from a higher database server that the attacker can obtain higher priority control ability. As a result, the application can suffer severe loss in providing normal services to its users or it may face total destruction. Such kind of crash of application can lead to bankruptcy of a company or a bank, even an industry. Sometimes the attacker can use such kind of attack to acquire confidential information that may be relevant to the security of a country. Consequently, SQL injection is very dangerous in many cases where the attacker makes use of the unchecked assumptions into the web application to obtain the unauthorized information [4].

There are 7 types of SQL injection that can be used by the attackers all are aiming to reform the SQL query statements [5]. In other hand, many researchers and practitioners are working to find a method that use to detect SQL injection attacks from crucial web applications [12].

II. SQL Injection

The hacker is using the SQL injection to attack any web site on the internet.. SQL injection attack the web from the page that the user can input the code the user login page can be used by the user to be able to gain the access of that website on which this attack is successful. The user login page takes the usernames and password by executing SQL Query as: SELECT * FROM user WHERE useeamil= 'M@mail.com' AND pswrd= 'mypwrd'; the weakness in this SQL query statement is that two input variables, i.e. username and password that the attackers can input the value directly without checking or filtering So the hacker can get the authentication of the admin by making use of malicious code as Username= ‘or’’=’ Password= ‘or’’=’SQL Query becomes like this: SELECT * FROM table_name WHERE username=“name“ AND password= „1“= „1“ OR „1“ = „1“ [6].



III. Related Work

[7] Shows a new approach. This research paper suggests some improvements to prevent web attacks. Authors believe that approach and measures suggest by us will help in securing the websites from code injection attacks in future.

[8]This process converts the input query into fruitful tokens on both client and server side and that are stored in a separate dynamic table. Both tables are compared, if they are different, a query is rejected and not forwarded to the database server. Otherwise, the query is proceeding further to the main database for retrieving result. It has better performance and provides increased security in comparison to the existing solutions. The goal of this paper is to provide improved security by developing a method which prevents illegal access to the database.

[3]Presents an authentication scheme for preventing SQL Injection attack using Advanced Encryption Standard (AES).Encrypted user name and password are used to improve the authentication process with minimum overhead. The server has to maintain three parameters of every user: user name, password, and uses secret key.

[9]This system is a web or non-web-based. It is distinguished by the multiplicity of its performing methods, so defense techniques could not detect or prevent such attacks. The main objective of this paper is to create a reliable and accurate hybrid technique that secures systems from being exploited by SQL injection attacks. This hybrid technique combines static and runtime SQL query analysis to create a defense strategy that can detect and prevent various types of SQL injection attacks. To evaluate this suggested technique, a large set of SQL queries has been executed through a simulation that had been developed. The results indicate that the suggested technique is reliable and more effective in capturing more SQL injection types compared to other SQL injection detection methods.

[20] Proposes a Static Analysis Framework in order to detect SQL Injection Vulnerabilities. SAFELI framework aims at identifying the SQL Injection attacks during the compile-time. This static analysis tool has two main advantages. Firstly, it does a White-box Static Analysis and secondly, it uses a Hybrid-Constraint Solver. For the White-box Static Analysis, the proposed approach considers the byte-code and deals mainly with strings. For the Hybrid-Constraint Solver, the method implements an efficient string analysis tool which is able to deal with Boolean, integer and string variables.

[10], proposed technique for detecting SQL injection is to dynamically mine the programmer-intended query structure on any input, and detect attacks by comparing it against the structure of the actual query issued. The Authors propose a simple and novel mechanism, called CANDID, for mining programmer intended queries by dynamically evaluating runs over benign candidate inputs. This mechanism is theoretically well founded and is based on inferring intended queries by considering the symbolic query computed on a program run. Their approach has been implemented in a tool called CANDID that retrofits Web applications written in Java to defend them against SQL injection attacks. The authors have also implemented CANDID by modifying a Java Virtual Machine, which safeguards applications without requiring retrofitting. They report extensive experimental results that show that our approach performs remarkably well in practice.

IV. Proposed system

In SQL injection the attacker can inject the SQL statements from the page that can be used to input his/her data from, like login page. Most of the hackers are using the text box tool to do this attack. From this text box. the hacker can access to SQL statements and inject the query. The hacker inserts user name or password. in such a way or forms to be likes like SQL query statement. The system will use these inputs as query statement. [8] [10] [13] [14] [15] [12]. Our goal of this method is to keep this input away from SQL statements. The system will call the database attributes in away that the user cannot put his /her input in query statement. For this purpose, we will convert this input to a string and compare with the data in the database as a string and this string cannot be used as query because it will not input or use in any query statement it will be statics statement use only to compare.

The sql query before attack or before run time

"select * from user1 where username ='" +user.Text + "' and password='" + pswrd.Text + "'"

If the hackers insert in pswerd.text the following inputs

User name= user

Password ="'or "1"="'1

The SQL will be

"select * from user1 where username ='" user "' and password='" "' or"1"="'1 "'"

The propose system will follow these steps

Convert user inputs to string

String user = user.Text

String paswrd= pswrd.Text

dr= "select * from user1" //here the dr store all the user name of application and here the attacker cannot injects the sql statement

for i= 0 to length of dr

if dr[i]==st

condition = true //

break
endif
endfor
The same will do with password also ,if both condition are true the system will check if the password and user name in same record by using this sql statement
"select * from user1 where username ='" +user.Text + "' and password='" + pswrd.Text + "'"
We can use this sql statement after we checked that the input of the user are not injecting the query statement
if the user input this value to system
Password =”’or “1”=”’1
String paswrd= pswrd.Text
Now the string paswrd =” ”’or “1”=”’1 ” this string will compare will all passwords in database.

The algorithm that follow by system:

1) Input the user name and password.
2) Convert user name to string (st1)
3) Call all the user names that stored in database and put them in one class (cl1)
4) Compare the user name string (st1) with all user name in class one by one
5) If user name found break , make condition1 true and go to 7
6) If the user name dose not found go to 15
7) Convert password to string (st2)
8) Call all the password that stored in database and put them in one class (cl2)
9) Compare the pass words string (st2) with all user name in class(cl2) one by one
10) If password found break , make condition2 true and go to 12
11) If the password dose not found go to 15
12) If both condition1 and condition2 are true go to 13 else go to 15
13) Check if both username and password from the same record if its true go to 14 else go to 15
14) Login is successful go to next page go to 16
15) Password or user name is not correct go to 16
16) End

The code below shows the full code of our method for login page by using asp.net

```
String user;
String password;
Int i,t=0,c=0,d=0,cond;
User= user.Text;
Password= pswrd.Text;
SqlCommand cmd= new SqlCommand("select
username from user1 ", con);
con.Open();
      SqlDataReader dr = cmd.ExecuteReader();     // dr
include all users name
For( i=0, i<=dr.length , i++)
{
   t=user.CompareTo(dr[i]);
   If (t==0)
     {
      c=1;
      Break;
     }                  //end of if
}                      //end of for
SqlCommand cmd2= new SqlCommand("select
password from user1 ", con);
      SqlDataReader dr2 = cmd.ExecuteReader();// dr
include all paswords
For( i=0, i<=dr2.length , i++)
{
   t2=user.CompareTo(dr[i]);
  If (t2==0)
  {
    d=1;
    Break;
   }                  //end of if
}                     //end of for
Cond=c+d;
If(cond==2) // that mean both password and user
name are true(c=1,d=1)
{
      SqlCommand cmd3= new SqlCommand("select *
from user1 where username ='" +user.Text + "'
      and password='" + pswrd.Text + "'", con);
      SqlDataReader dr3 = cmd.ExecuteReader();
     if (dr3.Read())
      {
         Session["username"] =
dr["username"].ToString();
         Response.Redirect("userpage.aspx");
      }
}
Con.close()
```

V. Results

The proposed method tested with default database includes 25 users, by this test all the possible statements that can be used for SQL injection have been used none of these string was able to pass the login page successfully, So the unauthorized users cannot get unauthorized access to the page. In this test the SQL injection statements have been used for user name and password text box. On the other hand all the authorized user can access their account successfully without any problem

VI. Conclusion

The weaknesses that the hackers exploit, is makes the input as SQL query statement and inject this SQL segment statement in SQL query statement to reshape the query statement. The web will consider this input as a query and not use to compare with data in a database. This system can prevent the SQL injection. It decomposes the SQL statement to parts and takes the result of each part alone, after that collect the result of each part to get the final result of the query (true or false). This method takes each condition of the SQL query statement and deals with it alone. The input of the user will be single unit and it cannot be decomposed to use as SQL query it will convert to string. The database attributes will compare with this string.

REFERENCES

1. Udit Agarwal , Monika Saxena, Kuldeep Singh Rana, " A Survey of SQL Injection Attacks" IJARCSSE, Vol 5, Issue 3, **2015**
2. Pooja Saini, Sarita, "Authentication Scheme to Protect Web Applications against SQL Injection AttackUsing Hash Functions" IJARCCE , Vol. 4, Issue 6**, 2015**
3. Ritu Gaur, Ravi Bhushan;" Protection against SQL Injection Attack on Web Applications Using AES and Stored Procedure", IJARCSSE, Vol 4, Issue 5, **2014**
4. ZhongdingDong1, Yun Liu,GuixunLuo and Sumeng Diao ," A Smart-driver Based Method for Preventing SQL Injection Attacks" , IJSIA , Vol.8, No.2 **, 2014**
5. Juhi Gupta ,Ruchi singhal ."SQL Injection A threads to web application" IJRCS , Vol. 2, Issue 1**, 2015**
6. Priyanka, Vijay Kumar Bohat, " Detection of SQL Injection Attack and Various Prevention Strategies" , IJEAT, Vol2, Issue-4**, 2013**
7. MANI SHARMA " MEASURES TO PREVENT WEB ATTACKS" A Peer Reviewed International Journal, Vol.3., Issue.3, **2015**
8. S.Anjugam and A.Murugan , " Efficient Method for Preventing SQL Injection Attacks on Web Applications Using Encryption and Tokenization", IJARCSSE, Vol 4, Issue 4**, 2014**
9. Jalal Omer Atoum and Amer Jibril Qaralleh "a hybrid technique for sql injection attacks detection and prevention" IJDMS , Vol.6, No.1 **2014**
10. Prithvi Bisht , P. MADHUSUDAN , V. N. VENKATAKRISHNAN "CANDID: Dynamic Candidate Evaluations for Automatic Prevention of SQL Injection Attacks" ACMTransactions on Information and System Security,Vol. 13, No. 2 **2010**.
11. X. Fu, X. Lu, B. Peltsverger, S. Chen, K. Qian, and L. Tao. "A Static Analysis Framework for Detecting SQL Injection Vulnerabilities",COMPSAC 2007, pp.87-96, July **2007.**
12. Bojken Shehu and Aleksander Xhuvani "A Literature Review and Comparative Analyses on SQL Injection: Vulnerabilities, Attacks and their Prevention and Detection Techniques" IJCSI, Vol. 11, Issue 4, No 1 **2014**
13. Neha Mishra," Sunita Gond Defenses To Protect Against SQL Injection Attacks" IJARCCE Vol. 2, Issue 10, **2013**
14. Jyoti Agrawal1, Mukesh Gupta DETECTION AND PREVENTION OF TAUTOLOGY AND UNION

QUERY BASED SQL INJECTION ATTACKS IJATES Vol 03, Special Issue No. 01 **2015**

15. Parveen Sadotra " A New Proposed Method for Detection and Prevention of SQLIA (SQL Injection Attack)" IJARCSMS Vol 3, Issue 5, May **2015**

